PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-141236
(43)Date of publication of application : 02.06.1995

(51)Int.CI. G06F 12/00
G06F 17/30

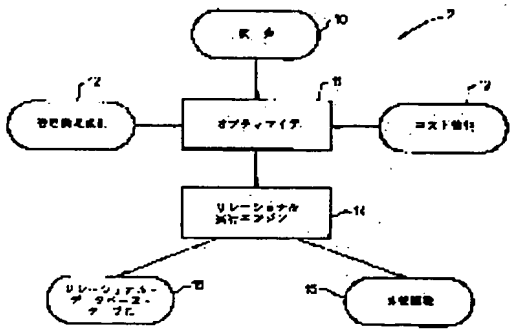
(21)Application number : 06-131601
(22)Date of filing : 14.06.1994

(71)Applicant : HEWLETT PACKARD CO <HP>
(72)Inventor : CHAUDHURI SURAJIT
SHIM KYUSEOK

(30)Priority
Priority number : 93 77227 Priority date : 14.06.1993 Priority country : US

(54) INQUIRY OPTIMIZATION METHOD AND DEVICE IN RELATIONAL DATA BASE SYSTEM PROVIDED WITH EXTERNAL FUNCTION

(57)Abstract:
PURPOSE: To efficiently and effectively execute optimization based on an inquiry cost to a relation inquiry in the case an external function is present by utilizing cost information and accessing a relational data base and the external function provided with a declaration type rewrite rule.
CONSTITUTION: A relational data base system 2 receives an inquiry 10 to be optimized and the inquiry 10 is supplied to an optimizer 11 for optimizing the inquiry 10 corresponding to a rewrite rule 12 and the cost information 13. The optimizer 11 receives the inquiry 10 to be processed, generates an alternate inquiry by using the rewrite rule 12 and further, selects an optimum plan from an inquiry group including the received inquiry and the alternate inquiry. In the relational data base system 2, a relational execution engine 14 capable of accessing a relational data base table 15 and the external function 16 is provided as well and the rewrite rule 12 is related to the external function 16.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平7－141236

(43)公開日　平成７年(1995)６月２日

(51)Int.Cl.[6]	識別記号	庁内整理番号	Ｆ Ｉ	技術表示箇所
Ｇ０６Ｆ　12/00	５１３　Ｄ	8944－5B		
17/30				
		9194－5L	Ｇ０６Ｆ　15/ 403　３４０　Ｄ	

審査請求　未請求　請求項の数１　ＯＬ　（全 18 頁）

(21)出願番号　特願平6－131601

(22)出願日　平成６年(1994)６月14日

(31)優先権主張番号　０７７２２７

(32)優先日　1993年６月14日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　590000400
ヒューレット・パッカード・カンパニー
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000

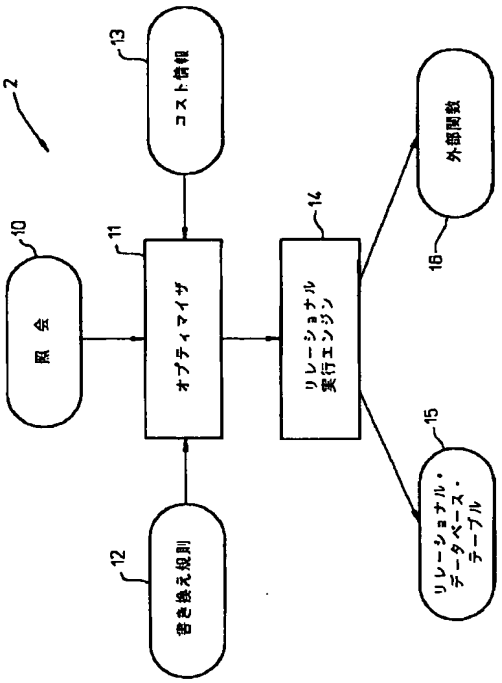
(72)発明者　スラジト・チャウドフリ
アメリカ合衆国カリフォルニア州94309スタンフォード，ピー・オー・ボックス・6870

(72)発明者　キュセオク・シン
アメリカ合衆国メリーランド州20770グリーンベルト，ナンバー101，ウエスト・ロード・168

(74)代理人　弁理士　古谷　馨　（外２名）

(54)【発明の名称】　外部関数を有するリレーショナル・データベース・システムにおける照会最適化方法及び装置

(57)【要約】
データベース・アプリケーションは典型的に外部関数(16)を呼び出すか、データベースにないデータにアクセスする必要がある。本発明はここで示した外部関数(16)における関係照会(10)をコストに基づいた最適化を行う包括的アプローチを提供する。この最適化はそのような意味を表現するための宣言型規則言語（例えばSQL）を使った外部関数について考慮されている。書き換え規則を適用する手順と同等照会の実行空間を生成する手順が記述される。この拡張された実行空間から最適プランを得る手順についても記述される。さらに、外部関数について必要とされるコストモデル拡張の必要性についても記述される。



【特許請求の範囲】
【請求項１】複数の論理レコードを有するリレーショナル・データベースと、関連する宣言型書き換え規則を有する少なくとも１つの外部関数と、
コスト情報を利用して、前記リレーショナル・データベース及び前記少なくとも１つの外部関数にアクセスする照会を最適化するためのオプティマイザから構成される、
リレーショナル・データベース・システム。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、リレーショナル・データベース・システムに関するものであり、とりわけ、外部関数を備えたリレーショナル・データベース・システムに関する照会最適化技法に関するものである。
【０００２】
【従来の技術】リレーショナル・データベース・システムはそのデータベースに記憶されたデータの照会を便利よく行えるようにする。しかし、多くの応用プログラムでは、データベースにとって外部のデータ及び操作（外部関数と呼ばれる）を統合する必要がある。例えば、関係照会の一部として、数学関数及びUNIXライブラリ関数を呼び出すのは便利である。さらに多くの問題領域に関して、高度に調整された応用プログラムが存在する。この応用プログラムを一から開発することはけた外れに高くつくので、こうした既存のプログラムを活用する能力は重要である。また、多くの応用プログラムでは、データベースには必要なデータの一部だけしか記憶することができない。それ以上のデータについては外部に常駐させることが可能である。外部データに対するアクセスは１組のインターフェイス・ルーチンによって可能になる。
【０００３】一例として、今日では地理データを記憶し、これにアクセスすることを可能にする多くの特殊な地理情報システム(GIS)が利用可能である。一方、属性（例えば、都市の人口）に関する情報はリレーショナル・データベースに記憶されるのが普通である。従って、GIS業務プログラムにとって関係照会言語を利用することのできる能力は、GISパッケージによって提供された関数を呼び出すことのできる能力と同じぐらい重要である。一般に、関係照会において外部関数を呼び出すことのできる能力は応用プログラムの開発において重要である。
【０００４】外部関数によって導入される最適化についての重要なチャレンジを明らかにするため、1992年にフロリダ州マイアミで行われたProceedings of the 3rd International Symposium on Large Spatial DatabasesにおけるKolovson他によるInteroperability of spatial and attribute data managers:A case studyに解説された初期の応用プログラムから取り上げた２つの例について考察することにする。この既知の応用プログラムはPapyrusプロジェクトにおいて作成されたものであるが、これについては、1991年にフロリダ州マイアミで行われたProceedings of the First International Conference on Parallel and Distributed SystemsにおけるConnors他によるThe papyrus integrated data server参照のこと。該応用プログラムは北カリフォルニアのベイエリアにおける事業及びその所在地に関する情報にアクセスすることを可能にする。該応用プログラムは、ETAK MapEngine及び関係記憶マネージャ上に作成された。カリフォルニア州メンロ・パークのETAK Inc.は車両用ナビゲーション装置を設計し、デジタル・マップ・データベースを作成する会社である。
【０００５】MapEngineは地理の記憶及び照会を可能にする地理データ・マネージャである。MapEngineはファイルMapにベイエリアにおける事業所の所在地を記憶している。リレーショナル・データベースはテーブルBusinessに事業に関する属性情報（例えば事業のタイプ）を記憶するために利用される。このテーブルの各論理レコードには、MapEngineに関するキーとしての働きをする追加属性も記憶される。MapEngineはこのキーを利用して事業所在地を検索する。同様に、MapEngineにおける各レコードはリレーショナル・データベースのテーブルBusiness内の対応する事業所に関する属性情報が含まれる論理レコードを指している。従って、照会範囲はMapEngine内だけでなく関係システムへと広げられる事になる。
【０００６】例１：この例の目的は最適化にとっての外部関数に関連した意味情報の重要性を強調することにある。MapEngineは地図における全てのポイント（さまざまな表現が可能であるが、ポイントまたはウインドウは単一引数として表現される）にアクセスするための関数(Map)、及び所与のポイントがウインドウ内にあるか否かをテストするためのブール関数(Inside)を提供する。関数Insideは算術関数である。MapEngineはウインドウが与えられると、そのウインドウ内にある地図の全ポイントを戻す関数Mapclipも提供する。ウインドウが与えられると、そのウインドウ内にある地図の全ポイントを見つける照会について考察することにする。この照会に対する応答は関数Mapを呼び出し、各検索位置がウインドウ内にあるかテストする（関数Insideを用いて）ことによって可能になる。しかし、関数Mapclipを利用すると照会の評価コストを大幅に減少させることができるので、関数Mapclipを呼び出すことによって照会に応答することができるという事実を利用することが重要である。
【０００７】例２：この例の目的は意味最適化によって所与の照会に修正を加える決定は照会のためのコストを考慮する必要があるという点を強調することにある。パロアルトのダウンタウンにある全レストランを見つける

問題について考察してみることにする。この照会に対する応答はテーブルBusinessから全てのレストランを選択し、次に関数Mapclipを実施することによって可能になる。しかし、MapEngineはレストランである全事業所から成るファイルMap_Restaurantも有している。従ってこの意味情報を用いて該照会に応答することができる。ファイルMap_Restaurantを使用してベイエリアの全レストランを求め、次に関数Insideを呼び出してパロアルトのダウンタウンにあるレストランを選択することが可能である。この２つの照会は同等であるが、パロアルトのダウンタウンに所在地を限定する索引付け効果が事業をレストランに限定することに基づく索引付けよりも有効であるか否かによって、たとえ程度問題であるにしろ、該照会の一方に関する最適プランのほうがもう一方の照会に関する最適プラン比べて優れている可能性がある。例１は、外部関数が存在する場合、同じ照会に応答するのに多くの方法が存在する可能性があり、こうした意味情報は照会の最適化にとって極めて貴重になるので十分理解する必要があるということを表している。例２は、照会最適化に関するこうした応用プログラムの意味情報は照会コストを考慮する必要があることを表している。

【０００８】関係照会に対する応答能力は評価オプションのレパートリ、及びこれらのオプションを選択するオプティマイザによって効果的に反映される。従って、関係照会によって外部関数を呼び出すことができる場合、データベース・システムはオプティマイザに十分な評価オプションと必要な拡張性とを提供して外部関数を伴う照会を有効に最適化できるようにする必要がある。しかし、既存のオプティマイザはこの要求を満たすことはできない。もちろん外部関数を呼び出す問題については他にも重要な点があるが（例えば、フォーマット変換、複雑なオブジェクトの支援）本発明では最適化と関連する問題だけに焦点を合わせている。

【０００９】近年、拡張可能度の異なるいくつかの拡張可能システムが提案されている。例えば、1990年12月のACM-SIGMOD RecordにおけるCarey他によるExtensible database management systems、1987年5月のカリフォルニア州サンフランシスコで行われたProceedings of the 1987 ACM-SIGMOD Conference on the Management ofDataの160-172ページにおけるGreafe他によるThe exodus optimizer、1989年6月にオレゴン州ポートランドで行われたProceedings of the 1989 ACM-SIGMOD Conference on the Management of Dataの377-388ページにおけるHass他によるExtensible query processing in starburst、1990年5月にニュージャージー州アトランティック・シティで行われたProceedings of the 1990 ACM-SIGMOD Conference on the Management of Dataの281-290ページにおけるStonebraker他によるOn rules,procedures,caching and views in database systems参照のこと。拡張可能システムにおける最適化のための書き換え言語も既知のところである。1992年5月に、カリフォルニア州サン・ディエゴで行われたProceedings of the1992 ACM-SIGMOD Conference on the Management of Dataの39-48ページにおけるPirahesh他によるExtensible/rule based query optimization in starburst、1988年6月にイリノイ州シカゴで行われたProceedings of the 1988 ACM-SIGMOD Conference on the Management of Dataの18-27ページにおけるLohmanによるGrammer-like functional rules for representing query optimization alternatives、Greafe他による上記論文参照のこと。Pirahesh、Lohman、及びGreafe他による参考文献の内容はここで参照することによって本明細書に組み込まれている。

【００１０】これら既知の書き換え言語の主たる欠点は、照会に現れる外部関数を扱う能力に欠けるということである。もう１つの欠点は、既存の書き換え言語が作業が厄介で複雑な中級プログラミング言語（例えば、Ｃ）に似ているということである。既存の書き換え言語のさらにもう１つの欠点は、最適化技法が書き換え規則によって左右されるので最適化が複雑になるということである。

【００１１】外部関数がある場合の照会最適化については、1989年8月にアムステルダムで行われたProceedings of the 15th International VLDB Conferenceの195-203ページにおけるChimenti他によるTowards an open architecture for LDLにおいて考察されており、この内容はここで参照することによって本明細書に組み込まれている。この参考文献においてLDLプログラムは外部テーブルを使用可能とし、外部テーブルに関するコスト記述子を定義するように拡張されている。LDLシステムは外部関数に関する意味情報を含む書き換え規則を利用していない。

【００１２】従って、外部関数を呼び出す照会を最適化しようとする従来の試みでは、書き換え規則に高級言語を利用して外部関数の意味関係を表そうとはしていない。また、従来の試みでは外部関数に関して書き換え規則及びコスト・モデルを利用して最適プランが得られることを保証することもできなかった。従って、こうした外部関数が存在する場合における関係照会の照会コストに基づく最適化に関する問題は、これまで満足のゆく取り組みがなされていなかった。

【００１３】

【発明が解決しようとする課題】本発明の課題は外部関数が存在する場合の関係照会に対する照会コストに基づく最適化に改良を施すことにある。

【００１４】

【課題を解決するための手段】一般的に言えば、本発明は外部関数に関する意味情報を考慮した最適化アプローチである。

【００１５】より詳しくは、本発明は外部関数が存在する場合の照会最適化に関する包括的なアプローチであ

る。書き換え規則を利用して外部関数の意味が表現される。書き換え規則は照会言語に対する拡張オプションを利用して指定される。書き換え規則はオプティマイザに対して他の同等な照会についての空間を示す。本発明によれば、さらに最適プラン（書き換え規則によって生成された代替照会についてのプランだけでなく、もとの照会についてのプランからの）が得られることが保証される。

【００１６】本発明には他に多くの態様があるが、これらについては実施例において完全な解説を行うことにする。例えば、本発明では書き換え規則の適用に関して、規則から独立した手順を利用して同等の照会を生成し、代替照会を最適化する場合に照会間における共通点を活用する。

【００１７】関係照会において外部関数を呼び出せることは、データベースにとって外部の既存のコード及びデータを活用する機会をもたらすので、多くの応用プログラムにおいて重要である。本発明によれば、外部関数を呼び出す照会が効率よく有効に最適化できるようになる。

【００１８】

【実施例】本発明は、同じ参照番号が同じ構成要素を示す図を用いた以下の詳細な説明により容易に理解される。

【００１９】以下では、図１～４を参照しながら本発明の実施例について解説する。ただし、当該技術の熟練者には明らかなように、これらの図に関し、ここで示される詳細な記述は説明を目的としたもので、本発明がこれらの実施例の範囲に限られるものではない。

【００２０】本発明は外部関数を呼び出す照会を最適化するための方法及び装置を具備したリレーショナル・データベース管理システムに関するものである。意味情報は、外部関数の意味を表現する書き換え規則を利用して、宣言として（SQLに対する単純な拡張オプションを利用した方法）指定される。意味情報を組み込むことによってオプティマイザに利用可能な選択の幅が暗黙のうちに拡張される。

【００２１】意味情報に含まれた全ての代替照会表現が考慮され、コストに対する考え方を基に最適プランが選択される。さらに、最適化アプローチには発見的方法を組み込むことが可能である。

【００２２】図１は、本発明によるリレーショナル・データベース・システム２を例示したブロック図である。リレーショナル・データベース・システム２は、最適化すべき照会10を受け付ける。この照会は書き換え規則12及びコスト情報13に従って照会10を最適化するオプティマイザ11に供給される。特に、オプティマイザ11は処理すべき照会を受け付け、書き換え規則12を用いて代替照会を生成し、さらに受け付けた照会及び代替照会を含む照会グループから「最適」なプランを選択する。リレーショナル・データベース・システム２には、リレーショナル・データベース・テーブル15及び外部関数16にアクセス可能なリレーショナル実行エンジン14も含まれている。書き換え規則12は外部関数16、またはリレーショナル・データベース・テーブル15に関連している。書き換え規則12の指定子によって書き換え規則の右辺及び左辺における照会が、全データベースにわたって同等であることが保証される。外部関数16は、データベースにとって外部のデータ及び操作を表示している（例えば、外部条件、外部テーブル、または外部関数）。

【００２３】図２は、本発明の態様を例示した基本フローチャートである。図２に示すように、本発明には３つの態様がある。第１の態様は外部関数に関する書き換え規則の適用20に関するものである。第２の態様は書き換え規則を用いた代替照会の生成22に関するものである。第３の態様はコストを考慮したアプローチを用いた最適プランの選択24に関するものである。これらの態様のそれぞれについて以下で詳述することにする。

【００２４】発明の第１の態様

本発明の第１の態様は、リレーショナル・データベース・システム２における外部関数16に関する書き換え規則12の適用20に関するものである。書き換え規則は外部関数（例えば、テーブル）を含む照会を、フォーマットが異なるが、やはり外部関数（例えば、テーブル）を含む同等の照会として書き換えるために用いられる。その書き替えられた照会はより効率よく実行することができる。従って、外部関数に関する書き換え規則は外部関数に関する意味情報を考慮することによって、照会オプティマイザの最適化潜在能力を高めることになる。

【００２５】本発明の焦点は論理積照会に向けられている。論理積照会は下記の形式をとるSQLの部分集合に対応する：

```
SELECT columnlist
FROM tablelist
WHERE cond1
AND ...
AND condn
```

WHERE節は、条件cond1...condnの論理積である点に注意されたい。論理積照会はどれも単純化された選択・射影・結合(SPJ)照会である。SQLのこの部分集合が最も広く用いられている。以下で述べる本発明の実施例は論理積照会に集中しているが、本発明は一般に全てのSQL照会に適用可能である。

【００２６】表記の便宜上、論理積照会は非再帰型Datalogで行う場合のように領域論理式で表現することができる。例えば、1989年のComputer Sience Pressから刊行されたUllmanによるPrincipals of Database and Knowledge-Bases Systems参照のこと。領域論理式の場合、論理積照会は被演算項（文字定数）の集合として表現される。各被演算項は引数を備えたテーブル名である。こ

うした領域論理式には明示的な等式で表される節はない。代わりに、等式は式における変数の一致として、暗黙的に表される。SQLと同様、こうした領域論理式の結果は論理レコードの集合である。このアプローチは意味の集合がこうした表記と連関している演繹データベースに用いられるアプローチとは異なっている。

【００２７】外部関数を参照するためには、領域論理表記における特殊な構文は必要ない。領域論理表記における外部関数に対する参照は単にもう１つの被演算項として示される。従って、外部関数は外部テーブルとしてモデル化される（外部関数及び外部テーブルという用語は交換可能である）。参照はSQL照会の場合、条件、テーブル、または関数として発生する可能性があるが、下記の例では、領域論理の表現において、照会の外部関数に対する参照がいかにして一様に被演算項として表現されるかが示される。

【００２８】例３：例１において簡単に述べた照会をわずかに修正したバージョンについて考察してみることにする。テーブルBUSINESSが５つの属性：NAME, TYPE, EARNING, SIZE, 及びETAKIDを持っているものと仮定する。Map Engineにおける地図は、属性ETAKID及びLOCATIONから構成される外部テーブルMAPとしてモデル化される。両方のテーブルにおける属性ETAKIDは、MapEngineにあるキーを参照する。例１から、関数Insideが、ポイントがウインドウ内にあるか否かをチェックする働きをするという点を想起されたい。従って、それは照会のWHERE節における条件として表現することができる。最後に、外部関数EXPECTED-REVENUEは入力引数としてレストランのサイズを選択し、レストランの平均期待収益を推定する。下記の照会は、ウインドウw内の地図の中にある全レストランを見つけ出し、どこが期待値を超える利益を得ているかを突き止める。

【００２９】SELECT
BUSINESS.NAME, MAP.LOCATION
FROM
BUSINESS, MAP
WHERE
BUSINESS.TYPE= ‘Restaurant’
AND
BUSINESS.ETAKID=MAP.ETAKID
AND
INSIDE(w, MAP.LOCATION)
AND
BUSINESS.EARNING>EXPECTED-REVENUE(BUSINESS.SIZE)
同じ照会に関する領域論理表現は：
Query(name):
-Business(name, “Restaurant”, earn, size, eid),
Map(eid, location),
$Inside^{bb}$(w, location),
Exp_Rev^{bf}(size, exp),
earn>exp である。

【００３０】外部関数がテーブル(MAP)として発生するか、あるいは関数(EXPECTED-REVENUE)として発生するかによって領域論理表記におけるその表現が変動することになる。すなわち、テーブルとして発生する外部関数はテーブルの属性のそれぞれについて引数を有しているが、関数としての外部関数は全ての入力引数位置に対して１つの引数位置を有し、全ての出力引数位置に１つの引数位置を有している。上付き文字は外部関数に関する安全制約条件を示すために用いられている。ｎ項の被演算項の場合、上付き文字は各引数位置毎に１つのｎ項のリストである。上付き文字ｂ（制約）は、外部関数を呼び出す前に、引数に値を受け渡さなければならないか否かを指示する。そうでなければ、上付き文字はｆ（フリー）である。例えば、被演算項Insideは両方の引数が制約されることを必要とする。簡略化のため、以下では全ての引数がフリーである場合、あるいは制約情報が関連していない場合には、上付き文字は省略される。

【００３１】領域論理表現の場合、外部テーブルであれ、あるいは内部（すなわち、内部記憶された）テーブルであれ、全てのテーブルが照会において同様に示される。そうではあっても、外部テーブルとデータベース・テーブルとの区別は照会の実行並びに照会の最適化にとって重要である。

【００３２】書き換え規則の目的は、外部テーブルに関連した意味情報、及びデータベース・テーブルとの関係を獲得することにある。書き換え規則の表現は、宣言型である。書き換え規則が宣言型のため、公式の意味を提供することができるだけでなく、所与の照会に対する規則の適用及び代替案の生成22に関して、規則から独立したアルゴリズムが促進されることにもなる。

【００３３】書き換え規則の表現は照会言語SQLに対する単純な拡張だけしか必要としない。概して言えば、書き換え規則はフォーマットREWRITE QUERY1 AS QUERY2を有しており、ここでQUERY1及びQUERY2は関係照会である。必要なのは照会の結果が、同じ内容（すなわち、列数）を備えていることである。重要なポイントは、こうした規則を適用すると、省略時解釈によってオプティマイザ11が所与の照会10に対する代替照会として新しい照会を生成するということである（新しい照会だけを考慮するように指定することもできるが）。いずれにせよ、所与の照会10及び書き換え規則によって生成された代替照会22からの最終的な照会の選択24はコストを考慮した最適化に基づいて行われる。

【００３４】下記の表記は書き換え規則に利用される。
$E_l(x,y) \Rightarrow E_r(x,z)$
式$E_l(x,y)$及び$E_r(x,z)$は、論理積式であり、それぞれ規則の左辺(lhs)及び右辺(rhs)と呼ばれる。変数ｘ，ｙ及びｚは順序づけられた変数の集合である。書き換え規則の両側で発生する変数ｘの集合は普遍変数と呼ばれ

る。さらに詳細に後述するように、書き換え規則によれば、普遍変数はそのままで、規則の左辺が規則の右辺によって置換され得る。さらに表記⇔を用いて２つの規則が同時に指示される（すなわち、双方向規則）。

【００３５】書き換え規則の第１の例として、例１で非公式に用いられている規則について考察することにする。この書き換え規則は領域論理表記として次のように表現することができる：

Map(eid, loc),
Inside(window, loc)
⇒ Mapclip(eid, loc, window)

Mapclipに関する安全制約条件が(ffb)である点に留意されたい。この書き換え規則において、変数eid、loc、及びwindowは全て普遍変数である。

【００３６】書き換え規則の第２の例として、例２では下記の規則が非公式に用いられた。

Business(name, "Restaurant", earn, size, eid),
Map(eid, loc)
⇔ Map_Restaurant(eid, loc)

この規則によれば、全レストランの所在地を得るためにはBusinessとMapの連結を選択することもできるし、あるいはETAKファイルMap_Restaurantを利用することも可能である。ここでは、eid及びlocは普遍変数である。

【００３７】書き換え規則の第３の例として、MapEngineの下記規則によれば、あるポイントが２つの所与のウインドウに属しているか否かを別個にチェックする代わりに、そのポイントがウインドウの共通部分に属するか否かをチェックすることが可能である。

【００３８】Inside(w1, point), Inside(w2, point)
⇒ Inside(w, point), Intersect(w1, w2, w)

この書き換え規則を利用することによって、複数のウインドウにおける全事業を見つけ出す問題が、ウインドウの共通部分における全事業を見つける問題へと軽減される。

【００３９】書き換え規則の第４の例として、下記の書き換え規則は照会最適化を促進する。適用時に、使用可能な索引が存在することをオプティマイザに示すことは有益なことが多い。例えば、所与のeidに関してMapに索引があるものと仮定する。この仮定に基づく書き換え規則は次のようになる：

Map(eid, loc)
⇔ $\mathrm{Mapwithid}^{bf}$(eid, loc)

Mapwithidに関する安全制約条件は、呼び出す前にeidの指定を必要とする、(bf)である。従来の最適化と全く同様に、索引の利用はコストの考慮に基づくものであり、従って所与の照会と上述の規則を用いることによって得られた照会の中から選択することが必要である。

【００４０】概して言えば、書き換え規則を利用し、左から右の規則を適用して所与の照会に対する代替照会が生成される。書き換え規則に関連した形式的意味論は２つの構成要素、すなわち同等性と方向性を有している。２つの照会は任意のデータベースに関して同じ論理レコードの集合を結果としてもたらす場合、同等と言える。

【００４１】まず、書き換え規則は下記の同等性を仮定する。書き換え規則が$E_l(x,y) \Rightarrow E_r(x,z)$の場合、任意のデータベースに関して下記に明示するように、照会Q_l及びQ_rによって同じ論理レコードの集合が得られる：

$Q_l(x) : -E_l(x,y)$
$Q_r(x) : -E_r(x,z)$

以上の同等性のため、書き換え規則を利用し、照会に部分式を「代入」することによって同等照会を導き出すことが可能になる。普遍変数だけが、Q_r及びQ_lの射影変数として生じるという点に注目されたい。

【００４２】次に、書き換え規則は矢印（⇒）で示すように、方向性も指定する。矢印は、規則の左辺のみが対応する同等の照会を生成するための右辺によって「置換され」（この逆はない）ることを指示するために用いられる。

【００４３】上述の書き換え規則の第１の例について考察してみることにする。意味論では任意のデータベースに関して、照会Q_l及びQ_rによる結果として同じ論理レコードの集合が得られなければならないことを暗示している。

【００４４】Q_l(eid, loc, window):
-Map(eid, loc),
Inside(window, loc)
Q_r(eid, loc, window):
-Mapclip(eid, loc, window)

この書き換え規則は、以下で示す照会Qにおいて生じる。規則の左辺がQの第２及び第３の被演算項に一致する。該被演算項の代わりに規則の右辺から対応する代入を行うと、照会Q'が得られる。

【００４５】Q(name, loc):
-Business(name, "Restaurant", earn, eid),
Map(eid, loc),
Inside(w, loc),
Intersect(w1, w2, w)
Q'(name, loc):
-Business(name, "Restaurant", earn, eid),
Mapclip(eid, loc, w),
Intersect(w1, w2, w)

意味論上の方向性は、書き換え規則の第３の例における規則が適用されると同等の照会が得られるが、照会Qには適用できないことを暗示している。

【００４６】本発明の第１の態様に関する望ましい実施例において、書き換え規則は照会言語SQLの拡張のような高水準の言語を用いてリレーショナル・データベース・システム２に提供する(20)。従って、上記においては論考を単純化するために領域論理表記が用いられている

が、書き換え規則は実際にはSQLの拡張記述の中で示される。例えば、上述の書き換え規則の第１の例は下記のように示すことができる：

```
REWRITE
SELECT eid,loc
FROM MAP
WHERE INSIDE(window,loc)
AS
SELECT eid,loc
FROM MAPCILP
WHERE MAPCLIP.WINDOW=WINDOW
```

発明の第２の態様

本発明の第２の態様は、書き換え規則16を用いて代替照会の生成(22)のために利用されるアプローチに関するものである。このセクションでは、単純化のため照会に不等式が含まれていないものと仮定する。

【００４７】書き換え規則の左辺に対して同等な部分式は新しい照会を導き出すために書き換え規則の右辺に置き換え可能であることに注意をすることが重要である。置換ステップは、簡単であるが、部分式が書き換え規則の左辺と同等であるか否かの判定は、もっと難しい。従って、２つの論理積照会の同等性をチェックする新規の手順について以下に述べることにする。

【００４８】従来の知識では、変数名称に変更が行われており、照会における文字定数が１対１に対応付けされている場合、及びこの場合に限って２つの論理積照会が同等であると信じられていた。換言すれば、２つの照会は同形の場合、及びこの場合に限って同等になる。しかし、書き換え規則の右辺を書き換え規則の左辺と同等であることが分かった部分式に置き換えるだけでは十分ではない。下記の例には、こうした単純な代入では、同等性を保証するのに不十分であることが示されている。

【００４９】例４：下記書き換え規則の左辺は照会Q_1に一致する。

```
Business(name, "Restaurant",earn,size,eid),
Map(eid,loc)
⇔ Map_Restaurant(eid,loc)
```

ここで、書き換え規則を利用して、Q_1から代替照会Q_1'を生成することができる。

```
Q1(loc):
-Business(bizname, "Restaurant",earn,size,eid),
Map(eid,loc),
Owner(bizname, "bob")
Q1' (loc):
-Map_Restaurant(eid,loc),
Owner(bizname, "bob")
```

Q_1'はQ_1に対する単純な代入アプローチを利用した結果である点に留意されたい。bobはモーテルだけしか所有していないものと仮定する。照会Q_1は空集合を戻すが、Q_1'は戻す必要がない。

【００５０】単純な代入に関する問題のポイントは、書き換え規則の意味論が普遍変数に対してだけ規則の両辺の同等性を保証するということにある。従って、対応付けが、１対１の対応付け条件を満たすだけでは不十分である。該対応付けは追加的な制約条件を満たさなければならない。従って２つの論理積照会の同等性をチェックするための新規の手順は、１対１の対応付けと同様に追加的制限も満たさなければならない。この新規の手順は対応付け代入と呼ばれる。

【００５１】対応付け代入は下記のように定義される。１⇒ｒが書き換え規則でQが照会であるとする。１の変数からQの変数への対応付けは、下記の場合規則から照会への対応付け代入と呼ばれる：

(a)対応付けが１の中の文字定数からQの中の文字定数に１対１で行われる；

(b)１の普遍変数だけがQの中の定数（もしあれば）に対応付けされる；

(c)非普遍変数のイメージがQの射影変数の中にない；

(d)１の非普遍変数のイメージが１のイメージにないQの文字として現れない；及び

(e)１の定数がそれ自体だけにしか対応付けられない。

【００５２】この定義において、対応付けに関する変数（または文字定数）の「イメージ」という用語は前者の対応付けが施される変数（または文字定数）を表している。同様に、対応付けに関する照会のイメージは照会に対応付けを施すことによって得られる文字定数を参照する。

【００５３】上記定義と例４において実施された対応付けを比較すると、この場合biznameが非普遍変数のイメージであるため、実施された単純な対応付けは条件(d)に背くことが判る。例４の照会の変形について考察してみると、文字定数OwnerはQにおいてHistoric(loc)に置き換えられる。こうした場合、条件(d)は満たされ、その対応付け代入が行われる。

【００５４】代替照会を生成するため、２つの基本的ステップを利用して書き換え規則が照会に適用される。まず、規則から照会への対応付け代入が必要な部分式が存在するか否かの識別を行う。次に、部分式を規則の右辺に置き換える（変数について名称変更をした後）。

【００５５】規則を照会に適用することによって得られる全ての同等の論理積照会を生成する手順は以下の通りである。１対１の対応付け条件を満たすことができる全ての対応付けが決定され、さらにその手順によって、該対応付けが対応付け代入の追加的制約条件を満たすか否かが判定される。単一テーブルへの照会の場合、規則の適用は、照会と書き替え規則のサイズ(n)における時間Ｏ(n log n)で実施することが可能である。実際にはほとんどの照会がこのカテゴリに含まれるので、規則の適用をしても大部分の照会についてはオーバヘッドがほとんど生じない。

【００５６】書き換え規則を導入する目的は、外部テーブルの意味によって生じる代替照会をオプティマイザ11に提供することである。代替照会は規則を適用することによって生成される(22)。生成された照会はそれぞれ所与の照会と同等である。

【００５７】照会の閉包は、書き換え規則の適用によって生成可能な全ての代替照会の集合を表している。すなわち、書き換え規則の集合Ｒに関する照会Ｑの閉包は照会の集合である：

閉包（Ｒ，Ｑ）＝｛Ｑ’｜Ｑ⇒*$_R$Ｑ’｝

記号Ｑ⇒*$_R$Ｑ’は、Ｑ’が、書き換え規則（集合Ｒから取り出される）を有限シーケンスで適用することによって、Ｑから得られたことを表すために用いられる。　書き換え規則は下記の例に示すようなシーケンスでも利用される。

例５：下記の照会について考察する。

Q(loc):
-Map(eid,loc),
Inside(w1,loc),
Inside(w2,loc)

書き換え規則の第３の例における規則を適用すると下記の照会Ｑ’が生成される：

Q’(loc):
-Map(eid,loc),
Inside(w,loc),
Intersect(w1},w2},w)

次に、書き換え規則の第１の例を適用すると下記の照会Ｑ”が生成される：

Q”(loc):
-Mapclip(eid,loc,w),
Intersect(w1},w2},w)

従来、最適化は照会に関する最適プランの選択に関するものであった。しかし、書き換え規則が存在する場合、本発明では同等照会の集合（閉包）が生成される。従って、最適化の問題は所与の照会の閉包における最適プランのうちから最も安価なプランを選択することにある。本発明の最適化アプローチは、最初に、同等照会の集合を生成(22)し（ステップ１）、次に得られた各照会の最適プランの中から最も安価なプランを選択(24)することである（ステップ２）。ステップ１については、詳細に後述し、ステップ２については、本発明の第３の態様において説明する。

【００５８】照会と書き換え規則の集合が与えられると、閉包手順によって書き換え規則の集合に関する照会の閉包が計算される。閉包手順については図３に示されている。

【００５９】図３は本発明の第２の態様による閉包手順の実施例に関するフローチャートである。閉包は入力照会と書き換え規則によって生成される同等照会の集合である。該手順は新しい照会が処理に利用可能か否かを判断(31)することによって開始される。最初のループ時にその新しい照会が入力照会10になり、利用可能になる。後続のループ時にはその新しい照会は、手順の他の部分によって生成された代替照会になる。

【００６０】いずれにせよ、利用可能な照会がなければ手順は完了する。一方、１つ以上の新しい照会が利用可能であると判定されると(31)次の新しい照会が得られる(33)。次に、少なくとも１つの書き換え規則が適用するために利用可能か否かに基づいて判定が行われる(34)。この判定34は少なくとも、例えば外部関数の呼び出しに少なくとも１つの書き換え規則が関連しているか否かを判定する。照会に用いるのに利用可能な書き換え規則がなければ、手順は判定31に戻り、次の照会の処理を行う。

【００６１】一方、少なくとも１つの書き換え規則が存在すると判定されると(34)、次の書き換え規則が得られる(35)。次の書き換え規則は、その照会に用いるために利用可能な書き換え規則の１つである。次の書き換え規則は、書き換え規則から照会への対応付け代入が実施可能であるか否かを判定する判定36において用いられる。対応付け代入の要件については上記において詳細に解説済みである。対応付け代入が失敗した場合は処理は判定34に戻り、別の書き換え規則が利用可能か否かが判定される。対応付け代入がうまくゆけば、新しい照会が書き換え規則（使用中の）及び照会（使用中の）によって生成される。その後、書き換え規則が使用中の照会（新たに生成された照会ではなく）に関して１つ以上の対応付け代入を有している場合、処理はループし判定36に戻る。

【００６２】特定の閉包アルゴリズム(gen_closure)に関するプログラミング・コードの一例が表１に含まれている。

【００６３】

【表１】

```
関数gen_closure(R,Q)
begin
  S=Q;
 δ=Q;
  repeat
    new=0
    for each q in  δ   and  r in R do
         new=new∪rewrite(r,q);
    endfor
    if new⊆S then returns(S);
    δ=new  - S;
    S=S∪δ ;
  forever
end

関数rewrite(r,Q)
begin
  Q_r=0
  for every sound application A
        of r to Q do
            Q_r=Q_r∪ {A_Q}
            ここで、A_Qはアプリケーションによって
            導き出された照会
  endfor
  returns(Q_r)
end
```

閉包計算アルゴリズム

```
手順OptPlan(Q):
begin
  if existsoptimal(Q) then return;
  Let Q={q_1,...,q_n};
  Let Si=Q- {q_i} ;
  for each i do
     OptPlan(S_i);
     Q_i=Plan for Q from S_i and q_i
  endfor;
  Q_iのうち最良のものを選択し、プラン・テーブルに
  追加する。
end
```

最適化アルゴリズム

【００６４】このアルゴリズムは、やはり表１に含まれている関数rewriteを繰り返し呼び出す。書き換え規則ｒ及び照会Ｑが与えられると、関数rewrite(r,Q)は、Ｑにｒを１回適用することによって得られる全ての照会を導き出す。表１において、「sound application」という句は対応付け代入処理の成功を表している。アルゴリズムgen_closureは繰り返し可能で、各繰り返し毎にrewriteを呼び出すことによって追加照会を生成する基となる働きをする照会の集合（新）が生じる。以前には生成されなかったこうして導き出された照会だけが、次の繰り返し処理の基となる働きをする。アルゴリズムgen_closureは、閉包を得るのに有効で、完全な手順である。

【００６５】関連する規則に有効にアクセスするには、規則の左辺に示される非演算項に索引付けされた規則テーブル（図４参照）に規則を保持することが望ましい。規則をクラス単位に分割することを可能にする、規則クラスを使うこともまた有効である。そして、指定されたクラスの全規則は他のクラスのどの規則よりも早く適用される。

【００６６】概して言えば、図３に示す閉包手順の場

合、終了は書き換え規則の集合に関する照会の閉包が有限か否かによって決まる。従って、該閉包が有限の場合、及びこの場合に限って該手順は終了する。しかし、書き換え規則の集合に関する照会の閉包が無限集合となるような照会、及び書き換え規則が存在しうる。結果として、閉包を促進するためいくつかの条件が識別される。

【００６７】書き換え規則の集合が、そのいずれかを満たせば閉包が保証される２つの条件について解説する。第１の条件は、長さ非増加特性条件である。この条件は、全ての規則の右辺の長さが、その左辺の長さ以下でなければならないということである。例えば、領域論理式の長さは式における非演算項の数と定義することができる。従って、書き換え規則の右辺がその左辺以下の長さであれば、その書き換え規則は長さ非増加条件を満たすことになる。例えば、下記の書き換え規則は長さ非増加条件を満たすことができない。

【００６８】Map_Restaurant(eid,loc)
⇒ Business(name,"Restaurant",earn,eid),Map(eid,loc)

第２の条件は、非空白特性条件である。長さ非増加特性条件は、たとえ長さ非増加特性条件の部分的な違反があっても、書き換えられた照会の長さを所定の限界を超えて増やすことができないという状況を正確に把握できない（すなわち、あまりに包括的すぎる）。非空白特性条件はこれらの状況を把握する。書き換え規則Ｒの集合の終了に関するこのテストには、２つのステップがある。まず、不等式の集合Ineq(R)が書き換え規則の集合Ｒから導き出される。不等式の集合Ineq(R)は次のように構成される：

－整数変数を全てのテーブル名に関連させる。

－全ての文字をテーブル名に対応する変数に置き換え、論理積を加算に置き換えることによって全ての領域論理式に関する代数演算式を得る。

【００６９】書式lhs⇒rhsの書き換え規則毎に算術不等式left_Exp≧right_Expが形成されるが、ここで、left_Exp及びright_Expは、それぞれlhs及びrhsに関する代数演算式である。さらに、全ての変数はゼロ以上である。書き換え規則の集合Ｒに対応する算術不等式の集合はIneq(R)で表示される。Ineq(R)の変数がいずれも空文字（ヌル）でなければ、書き換え規則の集合Ｒは照会毎に有限閉包を有している。Ineq(R)に対する解毎にゼロを割り当てなければならない場合、Ineq(R)の変数は、空文字（ヌル）であると言われる。線形計画技法に基づく効率の良いアルゴリズムが既知のところであり、線形不等式の集合における変数が全ての解においてゼロであるか否かをチェックするために利用可能である。

【００７０】非空白特性条件は、下記の例によって明らかにされる。長さ非増加特性を満たすことのできない、書き換え規則の第２の例におけるMap_Restaurantを含む、繰り返し適用される書き換え規則の集合について考察してみることにする。対応する不等式は（ｍはMap、ｒはRestaurant、ｄはMap_Restaurantとすると）：

｛ｍ≧０，ｄ≧０，ｒ≧０，ｄ≧ｍ＋ｒ，ｍ＋ｒ≧ｄ｝

明らかに空文字（ヌル）の変数はなく、従って有限閉包である。

【００７１】上記条件以外に、どちらの終了条件も満たされない場合（例えば、長さ非増加特性及び非空白特性）、閉包手順の終了を保証するために発見的方法を利用することも可能である。こうした場合、閉包手順は閉包の有限部分集合だけしか列挙しない。こうした部分集合を指定する技法については後述する。さらに、もし望むなら、発見的方法を利用して最適化手順のステップ２における考慮事項からステップ１で生成されたいくつかの照会を排除することも可能である。

【００７２】照会の閉包は、典型的に少数の照会に限定される。それにもかかわらず、発見的技法は最適化の第２段階において、最適化のために考慮される同等の照会の候補数を制限するのに有効である。これら発見的技法は、２つのタイプが可能である。まず、閉包の部分集合だけを列挙することが可能である。次に、最適化プロセスのステップ２における考慮事項から生成された代替照会のいくつかを排除することができる。

【００７３】閉包の選択的列挙においては、予算を使って列挙で費やされる最長時間を決めることができる。閉包手順は、この修正を組み込むために簡単に拡張することが可能である。もう１つの代替案は書き換え規則を修正することである。この代替アプローチについては下記の例によって説明する。

【００７４】例６：

Business(name, “Restaurant”,earn,size,eid),
Mapclip(eid,loc,window)
⇒ Business(name, “Restaurant”,earn,size,eid),
Mapclip(eid,loc,small_win),
Intersect(window,w,small_win)

といった書き換え規則は下記のように修正することが可能である：

Business(name, “Restaurant”,earn,size,eid),
Mapclip(eid,loc,window)
⇒ Business(name, “Restaurant”,earn,size,eid),
SpecialMapclip(eid,loc,window)

修正された書き換え規則には、新しいテーブル名SpecialMapclipが含まれている。SpecialMapclipが左辺に生じる書き換え規則はないので、もとの（無修正の）規則とは異なり、修正された書き換え規則は繰り返し利用することはできない。しかし、閉包を生成した後、

「SpecialMapclip(eid,loc,window)」

の代わりに、式

「Mapclip(eid,loc,small_win),
Intersect(window,w,small_win)」

が用いられることになる。

【００７５】従って、例に示すように部分式は単一の文字として扱うことができるので、有効閉包のサイズが縮小される。部分式の選択を利用して列挙のために選択される閉包の部分集合が決定される。この戦略を利用することにより、任意の書き換え規則の集合が長さ非増加特性条件を満たすことを保証し、その結果終了を保証することができる。

【００７６】発見的方法のもう１つの適用は最適化を必要としない同等の照会を識別することである。これは、大まかなコスト推定によって、あるいは下記アプローチを利用することによって判定される。書き換え規則の中には「always improving」のマーキングが施されるものもある（例えば、書き換え規則の第３の例）。こうした規則によって照会Ｑから照会Ｑ'が導き出される場合、Ｑ'によって常により良い最適プランが得られるものと仮定されるので、照会Ｑはもはや最適化の必要がない。

【００７７】発明の第３の態様

書き換え規則及び外部テーブルが存在する場合、照会の最適化は従来の最適化問題に対して新たな特徴を必要とする。まず、外部テーブルが存在するため、外部テーブルに関する評価を行うコスト・モデルの導入と新たな連結方法の導入が必要になる。このセクションでは、コスト・モデルが実行空間における所与のプランに実数を割り当て、動的計画法を利用する全ての関係オプティマイザにおいて暗黙の最適化の原理（1990年にMIT Pressから刊行されたCormen他によるIntroduction to Algorithms参照のこと）を満たすものと仮定する。さらに、外部テーブルは安全制約条件を有するので、従来の連結・列挙段階は安全制約条件を満たす連結の再配列だけしか考慮されないことを保証する必要がある。換言すれば、外部関数に受け渡される結合によって制約条件が満たされることを保証する必要がある。これは、十分に研究された問題である。例えば、検討に備えて1989年にComputer Science Pressから刊行されたUllmanによるPrincipals of Databaseand Knowledge-Base Systemsを参照されたい。最後に、該タスクは、複数で同等の照会を含む、拡充された集合から最適のプランを選択する(24)ことである。

【００７８】もちろん、最適性に関する検討は使用可能な集合に関して行わなければならない。従来より、照会の実行は、構文的には内部ノードが連結操作で、各葉ノードがベース・テーブルである、注釈付き連結ツリー（1986年8月に日本の京都で開催されたProceedings of International Conference on Very Large Data Basesの128-137ページにおけるKrishnamurthy他によるOptimization of nonrecursivequeries参照のこと）として表現される。本発明によれば、葉ノードも外部テーブルとすることが可能である。オプティマイザは左に深層をなす全注釈付きツリーを考慮するので、連結の線形配列が強制される。従って、実行空間は最適化プロセスのステップ１によって得られた各同等照会毎に、左に深層をなす全連結ツリーの空間であると定義される。次に、最適化プロセスのステップ２では実行空間から最小コストのプランを選択する。

【００７９】図４は、本発明による最適化プロセスの実施例に関するブロック図である。図４には図１に示されたオプティマイザ11に関連した機能単位が示されている。すなわち、最適化プロセスでは規則テーブル42に記憶された書き換え規則12を用いる同等照会ジェネレータ40を利用して、同等照会を生成する。最適化プロセスでは、また、再配列可能単位ジェネレータ44及び照会オプティマイザ46も利用する。照会オプティマイザ46は局所オプティマイザ47及びプラン・テーブル48との対話によって有効に機能することが可能である。本発明による最適化プロセスについては、さらに詳細に後述する。

【００８０】最適化プロセスのステップ２は、コスト最適化アルゴリズムを利用して実行空間から最適プランの判定または選択を行う照会オプティマイザ46によって実施される。以下では、コスト最適化アルゴリズムに関する２つの異なるアプローチについて解説する。

【００８１】コスト最適化アルゴリズムに関する第１のアプローチは従来の動的計画法最適化アルゴリズムの直接拡張である。実行空間は同等照会の実行空間の結合体であるので、閉包における各照会毎に最適プランを求め、次にそれらの中から最小コストのプランを選択することが可能である。例えば、Q_1及びQ_2が閉包内に２つだけしかない照会の場合、それぞれQ_1及びQ_2に関して最適なプランP_1及びP_2を判定し、次にP_1及びP_2のうちより安価なプランを選択することが可能である。従って、下記のコスト最適化アルゴリズムは従来の動的計画法アルゴリズムの直接拡張であり、次のステップが含まれる：

(i)従来の動的計画法アルゴリズムを利用して各照会を最適化し、その照会について最良のプランを求めるステップ。

(ii)ステップ(i)において得られた最良のプランの中から最も安価なプランを選択するステップ。

【００８２】ステップ(i)は、連結の順序を考慮している間に安全性をチェックするための直接的な拡張を用いて、従来の関係オプティマイザによって実行されることが可能である点に注目されたい（例えば1979年6月にボストンで開催されたProceedingsof the ACM SIGMOD International Symposium on Management of Dataの２３－３４ページにおけるＳｅｌｉｎｇｅｒ他によるAccess path selection in a relational database management of Data参照のこと）。このアルゴリズムに必要な実行空間は各同等照会の最適化に必要な実行空間の最大値である。しかし、該アルゴリズムは閉包の照会間に共通点があるにもかかわらず、この事実を利用することができ

ないので時間計算量に乏しい。

【００８３】コスト最適化アルゴリズムに関する第２のアプローチは直接拡張よりもはるかに速く機能するので望ましい。このアプローチのキーとなるのはコスト最適化アルゴリズムが最適化時に複数の同等照会全体にわたる照会間の共通点を活用するということである。従来の動的計画法に基づくアルゴリズムは一つの照会内における部分照会間の共通点を活用して最適化の時間計算量を軽減するが、最適化時に複数の同等照会全体にわたる部分照会間の共通点を活用することはない。

【００８４】改良型コスト最適化アルゴリズムの解説を行う前に２つ以上の同等照会によって共用されている部分照会を識別する方法について解説する。この識別プロセスは閉包内の同等照会が規則を適用することによって生成される22、最適化手順のステップ１において実施される。下記の例にはこの識別プロセスが示されている。

【００８５】例７：結果として照会Q'が得られる照会Qに対する書き込み規則の第２の例の適用について考察してみることにする。MapEngineのファイルHistoricには全ての史跡が含まれており、外部テーブルprice[bf](loc,amount)によって不動産的評価が得られる。以下の照会では史跡であってレストランのある場所に関する不動産的価格が求められる。

【００８６】Q(amount):
-Historic(loc),
Business(bizname, "Restaurant",earn,eid),
Map(eid,loc),
Price(loc,amount)
Q'(amount):
-Historic(loc),
Map_Restaurant(eid,loc),
Price(loc,amount)

部分式「Historic(loc),Price(loc,amount)」が、両方の照会において共通している点に注目されたい。この共通点は、書き換え規則の適用時に、その適用によっても変わらずに残っている最初と最後の文字定数を監視することによって検出することが可能である。照会Qの最適化の一部として共通部分照会Historic(loc),Price(loc,amount)に関する最適プランを決定しなければならないので、照会Q'に関する最適プランの決定時にそれを再利用することができる。

【００８７】改良型コスト最適化アルゴリズムによれば、各照会は１度に１つずつ最適化されるが、別の同等照会と共用される副照会の場合（該副照会に限って）、その最適プランはプラン・テーブル48に記憶されるので、その最適プランを再度導き出す必要はない。従って、最適プランを構成する場合にはプラン・テーブル48を調べて、そのプランが既に存在するか否かのチェックを行うことになる。

【００８８】改良型コスト最適化アルゴリズムに関するプログラミング・コードの例が、表１に示されている（すなわち、手順Optplan）。従来のアルゴリズムとは異なり、改良型アルゴリズムはトップ・ダウン・アプローチに従う。照会に非演算項が１つだけ、または２つだけしかないというOptplanの基本ケースは省略されている。改良型アルゴリズムは局所オプティマイザを呼び出して、S_i及びq_iからQ_iを生成する（表１参照のこと）。局所オプティマイザ47はコスト・モデルに関する情報を利用する。改良型コスト最適化アルゴリズムに従って、最適プランはボトム・アップで構築される。すなわち、ある照会の候補となる最適プランはすぐ下の副照会の最適プランから構成される。従来のオプティマイザのように、オプティマイザ11は組み込みブール条件（sargable述語）である、非演算項（例えば、salary>50k）を特別に取り扱う。それらは再配列されないが、その代わりその効果は局所オプティマイザにおけるアクセス経路の決定に関して考慮されると共に、選択要素に関しても考慮されることになる。最後に、改良型アルゴリズムでは安全制約条件（表１には示されていない）を満たさない連結配列を廃棄する。

【００８９】プラン・テーブル48にアクセスする場合、照会の最適プランの追求が効率良く行われることを保証するのが有益である。効率の良いアプローチの１つは、プラン・テーブルをハッシュ・テーブルとして保持することであり、そこには各照会が整数の分類済みリストとして表現されている。全ての整数が照会の文字が記憶されている汎用文字・テーブルの項目索引に対応する。従って、２つの照会の比較は２つのストリングが等しいか否かをチェックすることになり、従って、極めて効率よく実施することができる。下記の例には、文字の表現、並びにこの表現において照会間の共通点がいかに把握されるかが示される。

【００９０】例８：この例では、例７の照会Q及びQ'が利用されている。照会Qの文字定数に対応する汎用文字・テーブルは、左から右に1,2,3,及び4であると仮定する。従って、Qはストリング(1234)で表すことができる。しかし、書き換え規則が適用されると、新しい文字定数Map_Restaurant(eid,loc)が生成され、索引5と共に汎用文字・テーブルに入力される。Q'の表現は(145)になる。照会Qの最適化によって(14)に関する最適プランが生成され、プラン・テーブルに記憶される。照会Q'の最適化時には、最初にプラン・テーブルが調べられ、(145)に関するプランが既に存在するか否かの確認が行われる。存在しなければ、各副プランに関する最適プランが再帰的に構成される。すなわち、(14)に関する最適プランの構成前にプラン・テーブルが調べられ、(14)の最適プランが最適化のために再利用される。

【００９１】改良型コスト最適化アルゴリズムは副照会がなければ、副プランが再度導き出されるという望ましい特徴を有している。さらに、共用副照会に関するプラ

ンだけがプラン・テーブルに保持される。

【００９２】周知のように、動的計画法に基づくアルゴリズムはトップ・ダウン及びボトム・アップのいずれかで示すことが可能である（上記のCormen他による文献参照のこと）。従って、コスト最適化アルゴリズムの変形、すなわちボトム・アップ・アプローチによる変形が生成可能である。実際、ボトム・アップ・アプローチの場合、２つの可能性のある変形を生成することができる。可能性の１つは、全ての同等照会をまとめて最適化することである。従って、サイズｎの全ての副照会に関する最適プランを構成してから、サイズ(n+1)の任意の副照会に関する最適プランが構成される。このアプローチにはサイズ(n+1)の任意の副照会を構成するのにサイズｎのプランだけしか必要とせず、従って、空間の再利用が可能であるため、必要な空間がトップ・ダウン・アプローチよりも小さいという利点がある。あいにく、全ての同等照会に関する副照会がまとめて構成されるので、任意の同等照会に関する最適プランを完成するための時間はトップ・ダウン・アプローチに比べて長くなる。ボトム・アップ・アプローチのもう１つの変形では１度に１つずつ照会の最適化が行われるが、共用副照会の最適プランは保管される。これによって前述のアプローチに関する欠点は修正されるが、下記の例で示すように最大共通副照会を共用できないという問題が生じる。

【００９３】例９：閉包には(1234)及び(1235)と表現される２つの照会が含まれるものと仮定する。該照会は副照会(123)を共用している。(1234)にとって、従って(123)にとって最適のプランが既に構成されているものと仮定する。(1235)に関する最適プランを構築する作業について考察してみることにする。オプティマイザはボトム・アップ・アプローチを利用する場合、より小さいプランが全て構成されるまでキャッシュメモリに納められたプラン(123)の存在を認識することができず、従ってプラン・テーブルへの不必要なアクセスが行われることになる。

【００９４】従って、最適化アルゴリズムのボトム・アップまたはトップ・ダウン・バージョンの利用には賛否両論がある。分岐限定戦略で拡張されたトップ・ダウンの変形を利用するほうが望ましい場合もある。従って、ある部分プランが、それまで発見した最適プランのコストを超えたことが分かれば、最適状態に及ばないことが保証されるので、その部分プランを完成する必要はない。

【００９５】発見的技法を最適化プロセスに組み込むことも可能である。例えば、発見的方法を利用して照会が最適化される順序を決めることも可能である。また、所与の照会に関して最適プランをかなり上回る改善が得られた後では、最適プランの検索を止める（全空間を検索する代わりに）ほうが有益な場合もある。

【００９６】安価で密接に関連した述語は、やはり最適化において重要な役割を果たすことが可能である。従来の関係オプティマイザにおいて選択条件は、他のデータベース関係と同様、再配列されなかった。それどころか、選択条件はできるだけ早めに評価される。実際、再配列連結のコストは再配列される文字数の指数関数である。従って、不必要な再配列はオプティマイザにオーバヘッドをもたらすことになる。ポイントlocがウインドウw内にあるか否かのチェックを行うInside(w,loc)のような外部テーブルについて考察する。こうしたテーブルの呼び出しは安価である。従って、最適化コストはいくつかの外部テーブルを安価なテーブルとして指定することによって減少させることが可能である。安価なテーブルに対応する照会の文字は再配列されないが、残りの文字は再配列される。安価なテーブルに対応する文字は、再配列可能な文字の安全な配列で、できるだけ早めに評価される。安価な非演算項が存在すると、図４の再配列可能単位ジェネレータ44によって再配列可能な単位を生成するステップが導入される。関係オプティマイザにおけるsargable述語は記憶システム層にプッシュ・ダウン可能な特性を有している。上述のSelinger他による文献参照のこと。同様にどの安価なテーブルが再配列可能なテーブルに関連しているかを示す機能を与えることも可能である。最適化時には、この情報を活用することが可能である。

【００９７】次に、オプティマイザ46によって用いられる外部関数に関する照会処理技法について解説する。特に、焦点は連結操作のための照会処理技法に合わせられている。コスト・モデルに関する後続の解説では、外部テーブルが左に深層をなすツリーにおける連結ノードの右の子として生じるものと仮定する。従って、左に深層をなす連結ツリーにおいて、外部テーブルは左テーブルとして参照される他のテーブル（中間または基本）と連結されるものと仮定する。

【００９８】キー・ポイントは、外部テーブルに対するアクセスは安全制約条件を守らなければならないということである。従って、外部テーブルから論理レコードを得ることが可能になる前に安全制約条件のために必要になる設定値を受け渡さなければならない。多くのシステムでは、各設定毎に外部関数に対する呼び出しが行われる。こうした技法は呼び出しコストが高く、出力として多くの論理レコードを戻す外部関数の呼び出しには非効率的である。従って、本発明によれば、外部関数に関する照会処理は呼び出し段階と連結の残りを処理する段階の２つの段階を持つ外部テーブルとの連結を見ることから成っている。呼び出し段階は左テーブルから外部テーブルの設定引数に関する値を受け渡すことから構成される。この呼び出しを可能にする方法がいくつか存在する：

単純呼び出し：この案の場合、左テーブルの各論理レコード毎に安全制約条件の指定に従って対応する設定引数

に対する呼び出しが行われる。
グループ呼び出し：この案の場合、左の関係から設定引数のそれぞれ別個の値毎に設定引数に対する単一呼び出しが行われる。
【００９９】グループ呼び出し技法の場合、設定引数に関する別個の値の集合を識別するオーバヘッドが付加される。しかし、該技法には呼び出しが少なくて済むという利点があり、これは各呼び出しが高くつく（例えば、ある地点と他の全ての地点との平均距離を計算する）外部テーブルにとって重要である。さらに、左テーブルが、既に、連結前に、設定引数に基づいて分類されている場合には、グループ呼び出しのほうがよい。
【０１００】呼び出しによって論理レコードの集合が生成されるので、連結の残りを処理するステップは従来の連結と同様であり、任意の連結方法を利用することが可能である。また、外部テーブルの１つ以上のフリーな（ｆ）引数に当てはまる選択条件はこの段階で評価される。連結の残りを処理する段階にとって最も単純な選択は、各呼び出し毎の論理レコードの生成が内部テーブルとの結合として扱われる入れ子走査法である。唯一の相違は内部テーブルが各呼び出し毎に変化する（潜在的に）ことである。この連結の残りを処理する方法は、呼び出しに関する２つの技法と組み合わせることが可能である。
【０１０１】単純な呼び出しと入れ子走査結合技法の選択を組み合わせることによって従来の入れ子走査結合と同様の連結アルゴリズムが得られ、外部入れ子走査結合(FNL)と呼ばれる。グループ呼び出しと入れ子走査結合の選択を組み合わせることによって分類併合結合法と同様のアルゴリズムが得られ、外部分類併合結合(FSM)と呼ばれる。FSMアルゴリズムに関するプログラミング・コードの一例が表２に示されている（すなわち、外部分類併合結合アルゴリズム）。
【０１０２】
【表２】

```
関数FSM(Left,FTable)
(Leftは、左の関係、FTableは、外部テーブル)
begin
  Join=0
    Temp_Left=GROUPBY(Left,Bound)
    ここで、分類及びグループ化は、FTableの設定引数によって
    行われる。
    for every group Li of Temp_Left do
        FTi=Invoke(FTable,Bvali)
        ここで、Bvaliは、Boundの設定引数に関する区画Li
        における値である
        Join=Bag_Union(Join,Merge(Li,FTi))
    endfor
    return(Join)
end
```

外部分類併合結合アルゴリズム

【０１０３】FSMアルゴリズムは外部テーブルの呼び出しが高価な場合に望ましい。最後に、呼び出し数を減少させるため、呼び出し結果の把握がPostgresの文脈中に暗示されているのでこうした代替案を照会処理に対する当方のアプローチとともに利用することが可能である。例えば上述のStonebraker他による文献参照のこと。
【０１０４】コスト・モデルは所与のどのようなプランのコストでも計算できなければならない。従来の関係オプティマイザの場合、テーブルの記述子には各引数位置（すなわち、各列）における異なる値の数、及びテーブルにおける予想論理レコード数といったテーブルに関する統計的情報が含まれている。コスト・モデルは、記述子を利用して、操作（例えば、連結）のコストを計算する。また、コスト・モデルによって操作後に得られる中間テーブルの統計的情報を含む新たな記述子も生成される。
【０１０５】コスト・モデルに対する本発明のアプローチはデータベース及び中間テーブルに関する関係記述子を保存することであるが、本発明は２つの拡張を必要とする。まず、外部テーブルに関する記述子が与えられなければならない。次に、こうした記述子を利用して、ある操作のコストを求めることができるようにする方法、及び外部テーブルとの「連結」後に中間テーブルに関する関係記述子を導き出すことができるようにする方法に関する説明が与えられなければならない。
【０１０６】各外部テーブル毎に下記の情報を登録することができる：
安全制約条件：この情報は、コスト・モデルによって直接用いられるのではなく、オプティマイザによって用いられる。
コスト：外部テーブルを１回呼び出すコスト。
ファン・アウト：各呼び出し毎に予想される「出力論理

レコード」の数。
各属性毎に：
領域サイズ：各領域要素の表現サイズ。領域の制約（カーディナリティ）。カーディナリティに対して許容可能な割り当ては無限である。
異なる値の数：各呼び出し毎に属性が備える異なる値の数の期待値。このパラメータが明示的に与えられない場合、この係数を近似するためファン・アウトが利用される。全ての領域が有限の場合、一様分布が仮定され、この係数が同様に計算される。
【０１０７】安全性に関するものを除いて、記述子の他の全ての特性がこのコスト・モデルに関係している点に注目すべきである。該コスト・モデルは、1989年8月にアムステルダムで開催されたProceedings of the 15th International VLDB Conferenceの195-203ページにおける、Chimenti他によるTowards an open architecturefor LDLに提案されているモデルの拡張である。また、記述子のパラメータは、必ずしも定数である必要はなく、コンパイル時に照会の中に現れる定数に依存することも可能であるという点に注目すべきである。下記には、外部テーブルに関する記述子の一例が示される。
【０１０８】例１０：
外部テーブル$\mathrm{Intersect}^{bbf}$(window1,window2,window3)に関する記述子は、コストが0.012ms、ファン・アウトが１、異なる値の数の係数が１であることを特徴とする。各領域要素のサイズは実数に対応するサイズであり、領域サイズ（無限大を最大値とする）を有している。ファン・アウト及び異なる値の数が１である点に注目されたい。
【０１０９】記述子の計算には拡張が必要になる。簡略化のため、外部テーブルが左に深層をなす連結ツリーの右の子として生じる状況の解説だけしか行わない。外部テーブルは右の葉ノードとして生じるものと仮定されるので、左関係として参照される中間関係に関する記述子の存在を仮定することも可能である。オプティマイザ46の場合、結果を導き出す記述子の計算を行うため、カスタマイズされた関数を登録することが可能である。こうした関数は、引数として左関係の記述子を選択することが可能である。以下では中間テーブルに関する記述子を計算するための省略時の方法について解説する。
【０１１０】該記述子を導き出す公式について解説する前に、左の独自性係数について説明するのが有効である。左の独自性係数は左の関係に関する所与の記述子についての外部テーブルの別個の呼び出し期待数を推定する。左の独自性係数の近似方法がいくつか存在する。方法の１つは、以下で解説のように左の独自性係数に関する単純な公式を利用することである。
【０１１１】外部テーブルの場合、１つ以上の引数位置を設定することが必要になる。従って外部テーブルの設定引数に関して値を提供する左の関係の属性Ａに対応する集合が存在する。Ｐが左の関係における属性Ａの集合に関する異なる値の数であると仮定する。左の関係の記述子はＰの計算に用いられる。明らかに、個別の呼び出し数はＰを超えることができない。しかし、個別の呼び出し数は左の関係における論理レコード数であるＮを超えることもない。従って、min(P,N)は左の独自性係数として利用することが可能である。我々の公式が左の独自性係数の上限を提供するという点に注目されたい。該公式の適用の一例を示すことにする。
【０１１２】例１１：バス経路に関するターミナルの位置を提供する下記の照会について考察する。
Query(route,loc):
-Terminal(route,eid),Map(eid,loc)
Terminalに関する記述子は、100の論理レコードを有しているが、第２の引数における異なる値の数は10になると期待されるものと仮定する。こうした場合、左の独自性係数は10であると推定される。
【０１１３】以下の説明では、左のテーブルと外部テーブルの設定引数との間における一致以外に選択条件はないものと仮定する。出力選択条件の効果並びに記述子の外部テーブルに対する射影の影響は（連結ツリーの内部ノードと同様）左の関係と外部テーブルとの連結結果を中間テーブルとして取り扱うことによって計算される。従って、下記に示すコスト公式は呼び出し段階に限られる。
【０１１４】論理レコード数：連結の結果発生する推定論理レコード数は、$N'=F*N$であり、ここで、Ｆは外部テーブルのファン・アウト、Ｎは左の関係における論理レコード数である。
異なる値の数：外部テーブルのｉ番目の引数に対応する異なる値の推定数はUVF_i*UIであり、ここで、UVF_iはｉ番目の属性に関する異なる値係数である。パラメータUIは前述の左の独自性係数である。
コスト：外部入れ子走査法及び外部分類併合結合のコストが得られる。Ｎは左の関係の論理レコード数、Ｃは外部テーブルを呼び出すコスト、UIは独自性係数と仮定する。下記のコストは呼び出し段階だけのものである。
【０１１５】－　外部入れ子走査法：$C*N$
－　外部分類併合結合：$Cost_{sort}(N)+UI*C$
本発明の多くの特徴及び利点については以上の説明から明らかであり、従って、特許請求の範囲は本発明のこうした特徴及び利点の全てを網羅することを意図したものである。さらに、当該技術の熟練者であれば、容易に多くの修正及び変更が思い浮かぶことになるので、例示し、解説したそのままの構成及び働きに本発明を限定するのは望ましくない。従って、適合する修正及び均等物は、全て本発明の範囲に含まれるものとして考えることが可能である。
【０１１６】以下に本発明の実施態様を列挙する。
【０１１７】１．複数の論理レコードを有するリレーシ

ョナル・データベースと、関連する宣言型書き換え規則を有する少なくとも１つの外部関数と、コスト情報を利用して、前記リレーショナル・データベース及び前記少なくとも１つの外部関数にアクセスする照会を最適化するためのオプティマイザから構成される、リレーショナル・データベース・システム。

【０１１８】２．前記書き換え規則が、最適プランが選択される同等の照会を生成するために用いられることを特徴とする、前項１記載のリレーショナル・データベース・システム。

【０１１９】３．前記外部関数のそれぞれに関連した前記書き換え規則が高級宣言型言語で表現されることを特徴とする、前項１または２に記載のリレーショナル・データベース・システム。

【０１２０】４．前記書き換え規則に用いられる高級宣言型言語がREWRITE Query 1 AS Query2の書式を有するSQLの拡張であることと、書き換え規則が、書き換え規則の左辺に現れる被演算項に索引付けが施された規則テーブルに記憶されていることを特徴とする、前項１、２、または、３に記載のリレーショナル・データベース・システム。

５．前記オプティマイザが、複数の照会全体にわたって照会間における共通点を活用することと、前記リレーショナル・データベース・システムには、さらに共通点を活用するために、副照会に関して既に決定されている最適プランを記憶するプラン・テーブルが含まれることを特徴とする、前項１、２、３、または、４に記載のリレーショナル・データベース・システム。

【０１２１】６．コスト情報が、外部関数に関するコスト情報からなることを特徴とする、前項１、２、３、４、または、５に記載のリレーショナル・データベース・システム。

【０１２２】７．(a)外部関数に関する書き換え規則を提供するステップと、(b)実行すべき入力照会を受けとるステップと、(c)入力照会及び書き換え規則から代替照会を生成するステップと、(d)最適プランを生成するステップから構成される、リレーショナル・データベース・テーブル及び外部関数を呼び出す照会の最適化方法。

【０１２３】８．前記生成ステップ(c)が、対応付け代入を利用して、書き換え規則の少なくとも１つの左辺が照会の少なくとも一部と同等であるか否かを判定することからなることを特徴とする、前項７に記載の最適化方法。

【０１２４】９．前記生成ステップ(d)が、(d1)入力照会、及び代替照会の少なくとも１つに関して最適なプランを生成するステップと、(d2)最適プランの１つを選択するステップからなることを特徴とする、前項６または７に記載の最適化方法。

【０１２５】１０．前記生成ステップ(d1)では、複数の照会全体にわたって照会間の共通点を活用するということを特徴とする、前項９に記載の最適化方法。

【０１２６】

【発明の効果】本発明により、前記宣言型言語で表現される書き換え規則を簡単に記述することができ、外部関数を呼び出す照会について、コストを考慮した最適化が効率よく、かつ有効に実施される。

【図面の簡単な説明】

【図１】本発明によるリレーショナル・データベース・システムのブロック図である。

【図２】本発明の態様を示す基本フローチャートである。

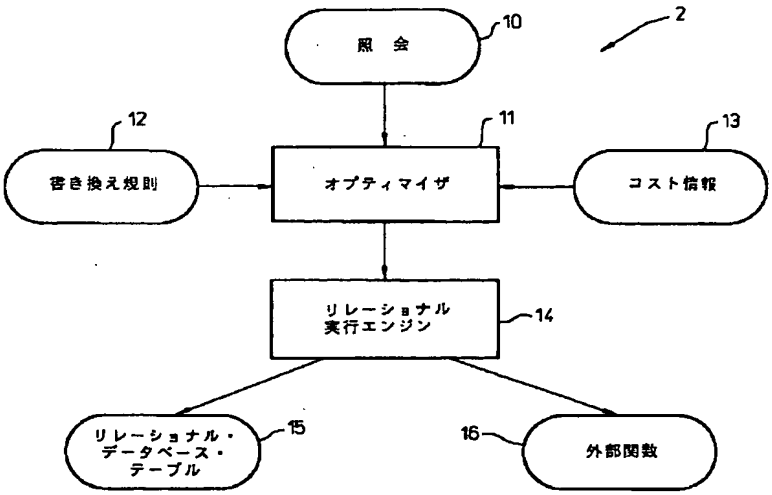
【図３】同等照会を生成する、本発明の第２の態様による閉包手順を示すフローチャートである。

【図４】本発明による最適化手順の実施例に関するブロック図である。

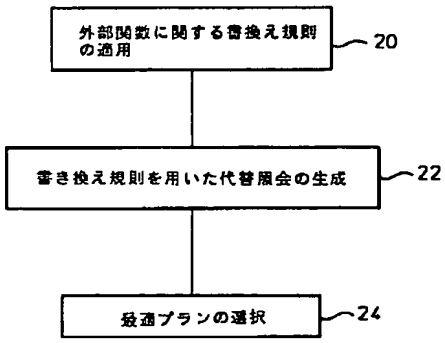
【符号の説明】

2　リレーショナル・データベース・システム
10　照会
11　オプティマイザ
12　書き換え規則
13　コスト情報
14　リレーショナル実行エンジン
15　リレーショナル・データベース・テーブル
16　外部関数
40　同等照会ジェネレータ
42　規則テーブル
44　再配列可能単位ジェネレータ
46　単一照会オプティマイザ
47　局所オプティマイザ
48　プラン・テーブル

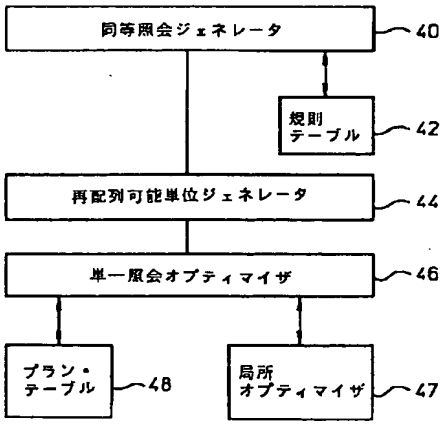
【図１】


照　会
10
2
12
11
13
書き換え規則
オプティマイザ
コスト情報
リレーショナル
実行エンジン
14
リレーショナル・
データベース・
テーブル
15
16
外部関数

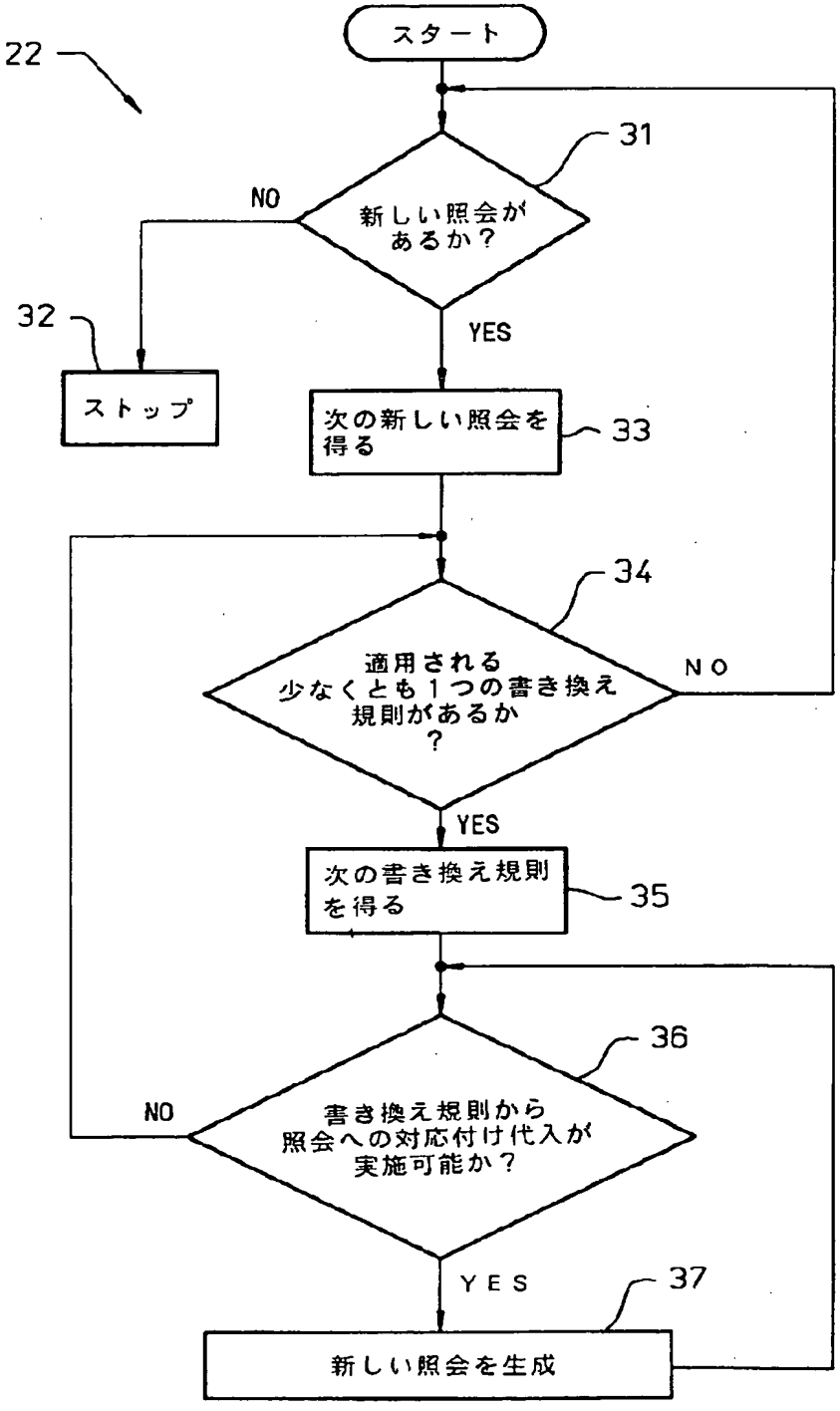
【図２】


外部関数に関する書換え規則
の適用
20
書き換え規則を用いた代替照会の生成
22
最適プランの選択
24

【図４】


同等照会ジェネレータ
40
規則
テーブル
42
再配列可能単位ジェネレータ
44
単一照会オプティマイザ
46
プラン・
テーブル
48
局所
オプティマイザ
47

【図３】


22
スタート
31
新しい照会があるか？
NO
YES
32
ストップ
次の新しい照会を得る
33
34
適用される少なくとも１つの書き換え規則があるか？
NO
YES
次の書き換え規則を得る
35
36
書き換え規則から照会への対応付け代入が実施可能か？
NO
YES
37
新しい照会を生成